METHOD AND CONTROLLER FOR PROGRAM CONTROL OF A COMPUTER PROGRAM HAVING MULTITASKING CAPABILITY

BACKGROUND INFORMATION

The present invention relates to a method for program control of a computer program, having multitasking capability, on a computing element, especially a microprocessor, of a

5   controller for controlling and/or regulating a system. The system may take up various possible system states.

The present invention also relates to a control program for program control of a computer program, having multitasking

10   capability, on a computing element, especially a microprocessor, of a controller for controlling and/or regulating a system. The system may take up various possible system states. The control program is capable of being run on the computing element.

15

Finally, the present invention also relates to a controller for the control and/or regulation of a system which may take up various possible system states. The controller includes a computing element, especially a microprocessor, on which a

20   computer program having multitasking capability may be run, and means for coordinating the execution of the computer program.

Conventionally, a computer program is used, for example, for

25   controlling and/or regulating a driving dynamics system (a so-called electronic stability program, ESP) of a motor vehicle. The driving dynamics system may take up various possible system states. Possible system states are, for instance, normal operation (ESP_normal), a first limited

operation (backup_ABS), in which a vehicle regulator (FZR) of the ESP is not functioning and only one antilock brake system (ABS) is functioning, a second limited operation (backup_EBD), in which only one system for the distribution of braking force

5   (electronic brake distribution, EBD) is functioning, in order to prevent at least over-braking the wheels on the rear axle, and a faulty state (fail-safe), in which all essential safety functions of the ESP, in particular FZR, ABS and EBD have failed. In order to avoid driving situations that are critical

10   to safety, the driver of the motor vehicle is notified, for instance acoustically or optically by the use of warning lights, of the various system states, but at least as regards the states in which only a limited or a faulty functioning still exists. The computer program is executable on a

15   computing element, which is particularly designed as a processor, of a controller for controlling and/or regulating the driving dynamics system.

The conventional computer program for controlling and/or

20   regulating the driving dynamics system is processed cyclically in a specified time pattern, i.e., only in a single time slice. The function references inside the computer program thus take place in a specified sequence, one after another. The sequence is specified so that the input variables of the

25   functions are available before the execution of the latter. In the case of input variables which are computed from other functions, these other functions thus have to be carried out first, before that function can be carried out which needs the input variables computed in the other functions.

30

In addition, conventionally, one may execute computer programs for controlling and/or regulating a system on an operating system having multitasking capability, and instead of processing the computer program in a single time pattern, one

35   may process it in various time patterns. This means, however,

that the functions of the computer program no longer have to be processed in a strictly fixed sequence, but that now other precautions have to be taken so as to ensure that the input variables of the functions are available before the execution of the latter.

Various priorities are allocated to the functions of the computer program. A higher priority is assigned to functions relevant to safety than to other functions. Higher priority functions are carried out in shorter time patterns, i.e., in time patterns which are repeated more frequently, whereas less safety-relevant functions are processed at a lower priority in longer time patterns which are repeated less often. It must particularly be ensured that the input variables of the functions are always present at the right point in time, i.e., a function which is processed, for example, in a 5 ms time pattern and requires input variables from a 40 ms time pattern, is permitted to be executed only after the 40 ms time pattern has been processed, and the required input variables have been computed.

SUMMARY

An example embodiment of the present invention coordinates the execution of a computer program, having multitasking capability, in such a way that the input variables required for carrying out a function of the computer program are present with certainty at the beginning of carrying out the function.

In one embodiment, a method is provided which includes:

- subdividing the computer program into a plurality of functionally linked functionalities;

- defining possible operating states for the functionalities;

- defining the possible system states of the system,
  in that the functionalities have operating states
  allocated to them that are specifiable for each
  system state;

5

- ascertaining of dependencies of the functionalities
  on one another, a first functionality being
  dependent upon a second functionality if at least
  one input variable of the first functionality is
10 ascertained in the second functionality; and

- centrally specifying the operating states required
  for a certain system state, taking into
  consideration the ascertained dependencies among the
15 functionalities and further boundary conditions.

An example method according to the present invention for
controlling the execution of a computer program having
multitasking capability may make it possible flexibly and
20 rapidly to control the computer program for controlling and/or
regulating a system in various time slices of an operating
system having multitasking capability by taking into
consideration the existing dependencies of the functionalities
among one another and further boundary conditions. Expanding
25 the computer program by the addition of further
functionalities is possible without any problem.

Thus, according to an example embodiment of the present
invention, the computer program is subdivided into a plurality
30 of functionally linked units, so-called functionalities.
Permissible operating states are defined for each of the
functionalities. The definition of the permissible operating
states may be made, for example, based on practical
experiences, based on simulations or based on theoretical
35 considerations. Moreover, those system states are defined

which the system is able to take up. Each system state is characterized by the fact that at least one functionality has a certain operating state, i.e., that at least one of the operating states takes up a specifiable value. For each system state, certain values are allocated to the operating states for the definition of the possible system states.

The functionalities of the computer program are subdivided into mutually dependent functionalities on the one hand, and into mutually independent functionalities on the other hand. Thus, for example, a first functionality is a function of a second functionality when at least one input variable of the first functionality is ascertained in the second functionality. The ascertained dependencies are stored in the controller, so that the ascertained dependencies are able to be accessed within the framework of the control of the execution of the computer program. For, according to the present invention, these dependencies of the individual functionalities on one another determine decisively the point in time at which a functionality is able to change from one operating state to another operating state, and thus, when the overall system is able to pass from one system state over into another system state.

As soon as all input variables of a certain functionality, by which a certain system state is characterized, are present, the operating state of this functionality is able to be changed. This applies also to all further functionalities by which the system state is additionally characterized. Only when all functionalities have been changed to the corresponding operating state (and when possibly further specifiable boundary conditions are fulfilled), the system enters into the desired system state. Expressed in other words, the desired system state, or the operating states of the functionalities which characterize this system state, are

specified centrally only when the ascertained dependencies
between the functionalities and further boundary conditions
have been satisfied.

5   According to one advantageous further refinement of the
present invention, an operating state is defined by an
operating state variable which is able to take up various
operating state values. Thus, each functionality has an
operating state variable allocated to it, which takes up
10   different operating state values depending on the operating
position. Depending on which operating state value the
operating state variable of a functionality has, the
functionality is available, for instance, in complete measure,
in limited fashion or not at all.
15

The various operating states of a functionality may be
conditional upon the fact that not all required input
variables are available as yet, when the functionality is
carried out. If, in such a case, it should not be possible to
20   carry out corresponding functionalities for the computation of
these input variables ahead of time, the functionality is
executed using input variables which were ascertained with the
aid of an alternative algorithm or by modeling from other
variables. However, this may lead to the functionality not
25   being available in full measure but only in a limited way. In
case the missing input variables are not able to be modeled or
otherwise computed, it may happen that the functionality is
not available at all.

30   According to one example specific embodiment of the present
invention, the operating state variable may take up operating
state values corresponding to the settings "full
functionality", "limited functionality" and "no
functionality".
35

For taking into consideration the ascertained dependencies among the functionalities, the processing of the functionalities which characterize a certain system state may be staggered in time in such a way that the second

5   functionality is processed before the first functionality, so as to ascertain the at least one input variable for the first functionality.

Alternatively, for taking into consideration the ascertained

10   dependencies among the functionalities, the at least one input variable for the first functionality is not ascertained by processing the second functionality, but in another way. The other way of ascertaining the at least one input variable for the first functionality advantageously includes a modeling of

15   the input variable from other variables, ascertaining a substitute variable and/or ascertaining the input variable with the aid of an alternative algorithm.

According to another advantageous further refinement of the

20   present invention, the operating states may be specified by functionalities characterizing a certain system state as a function of at least one fault that has occurred in the system. Thus, according to this further refinement, fault conditions in the system are detected and localized. The

25   faults that have occurred in the system have an effect on the selection and specification of a certain system state. Those input variables, on which the fault condition has an effect, are indicated correspondingly (input variable status signal). The selected system state and the status of the input

30   variables is then taken into consideration for the selection and specification of the operating states of the individual functionalities which characterize the specified system state. A strategy is developed as to how the individual functionalities may be carried over into the operating states

35   required for the selected system state, while taking into

consideration the ascertained dependencies among the functionalities. When all the functionalities characterizing the system state have the required operating states, the transition of the system into the specified system state is finished.

According to another advantageous further refinement of the present invention, the operating states may be specified by functionalities characterizing a certain system state as a function of the actual operating states of the functionalities. According to this further refinement, the instantaneous actual system state is ascertained with the aid of the actual operating states of the functionalities. In addition, the actual operating state is taken into consideration during the control of the operating states among one another so that, for example, a new setpoint operating state is specified only when at least one other functionality has taken up a required operating state. In a driving dynamics system of a motor vehicle, for example, a new setpoint operating state is specified only when the operating state of an ABS controller has completed the transition of a limited to a full operativeness.

In addition, according to the present invention, two particularly advantageous uses of the example method according to the present invention for controlling the execution of a computer having multitasking capability are provided. On the one hand, using the method for controlling the execution of a computer program for controlling and/or regulating a system in a vehicle, especially a motor vehicle, is provided. On the one hand, using the method for controlling the execution of a computer program for controlling and/or regulating a driving dynamics system in a motor vehicle, is provided. On the other hand, using the method for controlling the execution of a computer program for controlling and/or regulating a system in

a building is provided. In particular, using the method for controlling the execution of a computer program for controlling and/or regulating an alarm system, a heating or air conditioning system and/or an access control system in a

5   building is proposed.

Particularly significant is the implementation of the method according to an example embodiment of the present invention in the form of a control program that is provided for the control

10   and/or regulation of a system which is able to take up various possible system states. In this context, the control program is executable on a computing element, particularly on a microprocessor, of the controller and is suitable for carrying into effect the method according to the present invention. In

15   this case, therefore, the present invention is implemented by way of the control program, so that the control program constitutes the present invention in the same way as the method for whose execution it is suitable. In this connection, it is preferred if the control program is stored on a storage

20   element, especially on a read-only memory, a random access memory or on a flash memory.

Furthermore, an example controller according to the present invention may also include:

25

     -     means for subdividing the computer program into a plurality of functionally linked functionalities;

     -     means for defining permissible operating states of

30   the functionalities;

     -     means for defining the possible system states, these means allocating specifiable settings to each system state;

35

- means for ascertaining of dependencies of the functionalities on one another, a first functionality being dependent upon a second functionality if at least one input variable of the first functionality is ascertained in the second functionality; and

- means for centrally specifying the operating states required for a certain system state, taking into consideration the ascertained dependencies among the functionalities, and further boundary conditions.

According to one advantageous further development of the present invention, the controller may have means for carrying out the method according to the present invention.

BRIEF DESCRIPTION OF THE DRAWINGS
Further features, uses and advantages of the present invention come to light from the following description of exemplary embodiments of the present invention which are shown in the drawings. In this context, all of the described or represented features, alone or in any combination, form the subject matter of the present invention, regardless of their formulation and representation in the specification and drawings, respectively.

Figure 1 shows various system states of a system.

Figure 2 shows a functionality of a computer program having multitasking capability for controlling and/or regulating a system.

Figure 3 shows a flow diagram of an example method according to the present invention.

Figure 4 shows a controller of the present invention according to one example embodiment.

DESCRIPTION OF EXAMPLE EMBODIMENTS

The present invention relates to a computer program having multitasking capability for controlling and/or regulating a system. The computer program is executable on a computing element, particularly on a microprocessor, of a controller for controlling and/or regulating the system. The computer program having multitasking capability is processed in several different time patterns. The individual time patterns repeat cyclically, but viewed on the whole, the computer program is not processed cyclically.

The computer program is subdivided into a plurality of tasks, to which various priorities are allocated. Higher priorities are assigned to tasks having safety-relevant objectives than to tasks which have no safety-relevant objectives. The higher priority tasks are carried out in shorter time patterns, i.e., they are processed more frequently per time unit than the lower priority tasks.

The subdivision of the computer program into a plurality of tasks relates to the software-technological implementation of the computer program. On the functional plane, the computer program is subdivided into a plurality of functionally linked units, so-called functionalities. A functionality is able to include one or a plurality of tasks. In the case of a computer program for controlling and/or regulating a driving dynamics system (electronic stability programs, ESP) in a motor vehicle, functionalities are processed such as an antilock brake system (ABS), which prevents locking of the wheels during braking, or an automatic vehicle controller (FZR), which undertakes well-directed braking interventions at individual wheels, in order to maintain the driving dynamics

of the motor vehicle.

On account of the fact that, in a computer program having multitasking capability, the function requests do not simply follow one another, and thus it cannot be simply ensured by the sequence of the requests that the input variables of a functionality have already been ascertained by a previously executed functionality, in the case of computer programs having multitasking capability, other precautions are taken so as to ensure that the required input variables are always correctly present for the functionalities to be executed. Thus, a functionality which is called up in a 5 ms task and which needs input variables from a 40 ms task, at the first call is executed only if the 40 ms task had already been computed.

One aspect of the present invention is that transition conditions are assigned to each of the system states or rather to each (permissible) transition from a first system state to a second system state, and the execution of the computer program is controlled in such a way that the system is carried over into the second system state only when all the transition conditions assigned to the transition into the second system state have been satisfied. If, for example, the transition condition is that all input variables are available to a functionality characterizing the second system state, then, in the light of the method according to the present invention it may be ensured that the overall system is indeed only carried over from the first system state into the second system state when all the required input variables are present.

The various system states 30 are shown in Figure 1 by the example of a driving dynamics system (ESP). Among others, the following system states 30 are possible:

-       "full system": normal operation, full operability of the driving dynamics system;

-       "backup_ABS": antilock brake system (ABS) only, no vehicle controller (FZR) is active, limited operability;

-       "backup_EBD": only electronic brake distribution (EBD) active, limited operability;

-       "failsafe": FZR, ABS, EBD inactive, no operability of the system of any kind; and

-       "XYZ": any other system state.

The transitions between the system states are marked with reference numeral 31.

In Figure 2 a functionality X is shown, which has the input variables in_i and the output variables out_i. Between input variables in_i and output variables out_i a slider is provided that is able to be moved in the horizontal direction, which represents three different operating states A, B, C of functionality X. By sliding the slider, operating state A, B, C of functionality X may be changed.

The various system states 30 of the system are characterized by the fact that at least one of functionalities X of the system has a specifiable operating state A, B, C. The sum of operating states A, B, C of functionalities X thus yields corresponding system state 30 of the overall system. To each functionality X of the computer program an operating state variable is assigned, which may take up various operating state variables which respectively correspond to a certain operating state A, B, C of functionality X.

Switching a functionality X into another operating state A, B, C may, for example, be required when not all input variables in_i, required to execute functionality X, are present. At first one may try to delay the execution of this functionality X to the point that all required input variables in_i are present, i.e., until other functionalities X, in which the required input variables in_i were ascertained, have been executed. However, situations are possible in which delaying the execution of a functionality X, until all required input variables in_i are present, is not possible. In such a case, missing input variables in_i may also be modeled in the light of other variables, or may be computed by using an alternative algorithm. It is also possible, instead of the missing input variable in_i, to draw upon another variable which is already available, for executing functionality X. All these measures, however, which may be resorted to in case a required input variable in_i is not available, in the last analysis lead more or less to a limitation of the operability of the functionality, which is expressed by a change in operating state A, B, C.

In Figure 3, a method is described for controlling the execution of a computer program having multitasking capability, for instance, in the light of a driving dynamics system in a motor vehicle. This method, and particularly the present invention may, however, be used for any system that is controlled and/or regulated by a computer program having multitasking capability. A further possibility for use, which is here addressed explicitly, is the use of the method according to the present invention for controlling the execution of a computer program for controlling and/or regulating an alarm system, a heating and air conditioning system and/or an access control system in a building, that is, the use of the method according to the present invention in the field of building management.

In a function block 1, the so-called platform software (PSW) is monitored. "Platform software" is the low-level part of the computer program for controlling and/or regulating the driving dynamics system. In a function block 2 the user software (ASW) is monitored. "User software" is, in the case of a driving dynamics system, the ABS regulation or the vehicle controller (FZR). Function blocks 1 and 2 are used to detect faults 11, 12 in the respective software components. A possible fault 11, 12, which could be detected in function blocks 1 and 2, would be a sensor fault which prevents a certain input variable in_i, which is required for the computation of a functionality X, from being available.

Faults 11, 12, detected in function blocks 1 and 2, are transmitted to a function block 3 which is implemented as a macro. In function block 3, in the light of faults 11, 12 ascertained in function blocks 1 and 2, a corresponding fault condition 13 of the system is determined. This fault condition is transmitted by function block 3 to a further function block 4, which is denoted as a failure processing system (FPS). While taking into consideration the ascertained fault conditions 13, in function block 4, an appropriate strategy for transition into the second system state, or more exactly, a strategy for targeted change of operating states A, B, C of the functionalities characterizing the second system state, is determined. Strategy 14, determined in function block 4, for switching operating states A, B, C of functionalities X to transition into the second system state, is transmitted to a function block 5. Stated more exactly, according to determined strategy 14, one after another, various setpoint operating states are transmitted, from function block 4 to function block 5, of those functionalities which characterize the second system state. Determined strategy 14 thus represents a setpoint system state, in the present exemplary embodiment the second system state.

Fault conditions 13, determined in function block 3, are also transmitted to a function block 6, in which the status of input variables in_i of functionalities X are characterized by setting a so-called invalid bit. For each input variable in_i of functionalities X of the computer program, its own status signal in the form of the invalid bit is provided. Thus, when in function blocks 1 or 2 a sensor fault 11, 12 has been detected, those input variables in_I, which are impaired by sensor fault 11, 12, are characterized appropriately by the setting or deleting of the invalid bit. Status signal 15 is also transmitted to function block 5.

In a function block 7, actual system state 16 is ascertained and is also transmitted to function block 5. As an example of a driving dynamics system, actual system state 16 includes the state of the driving dynamics system per se, but also the driving state of the motor vehicle. In a function block 8, the dependencies 17 of operating states A, B, C and the functionalities X are ascertained among one another. The dependencies 17 ascertained are also transmitted to function block 5.

In function block 5, setpoint operating states 18 are prepared as a function of the variables 14, 15, 16, 17 obtained from function blocks 4, 6, 7, 8. In particular, it is checked in function block 5 whether the operating state variables of functionalities X have the operating state values required for the second system state, i.e., whether the functionalities characterizing the second system state are in the required operating states. Function blocks 4 through 8 are summarized in a superordinated function block 9, which is denoted as a controller release system (CRS).

In case it is determined in function block 5 that the operating state variables have the required operating state

values, i.e., that the functionalities, which characterize the second system state, are in the required system state, function block 5 specifies one or a plurality of setpoint operating states 18 and transmits these to a function block

5   10. In function block 10, the user software (ASW) and a security software (SIS) as well as an offset preparation is contained. The ASW corresponds to the controller part of the software (e.g. to the ABS, ASR or engine torque controller). The corresponding functionalities in function block 10 are

10   then switched to setpoint operating state 18. Actual operating state 19 is transmitted from function block 10 to function blocks 7 and 8. There they are drawn upon for ascertaining the actual system state in function block 7 and for ascertaining the dependencies of functionalities X among one another in

15   function block 8.

During the transition from one operating state A, B, C to another, in principle, two different kinds of transmission may be distinguished:

20

-    the transition from an operating state of a lower priority to an operating state of a higher priority as, for example, the transition from ABS_full system to ABS_off. This transition takes place directly, so

25   that further, possibly faulty controls are not able to occur.

-    the transition from one operating state of a higher priority to an operating state of a lower priority as, for example, from ABS_off to ABS_full system. In

30   this case, the transition from the setpoint operating state to the actual operating state is determined by the functionality itself. In this context, the actual operating state must remain

35   fully functionable until the setpoint operating

state has been reached. During the switch-over phase
from the actual operating state to the setpoint
operating state, both operating states are computed
in parallel. Thus, the functionality itself
5   determines when the transition is to take place. It
should be noted that, during the transition phase,
acoustical or optical warning instructions must
further be output. On the part of the FPS (failure
processing system) in function block 4, warning
10   instructions are no longer output, since by one
reset, faults 11, 12 detected in function blocks 1
or 2 were already reset.

In Figure 4, a controller according to the present invention
15   is designated in its entirety by reference numeral 20.
Controller 20 is used for controlling and/or regulating a
system which can take on various possible system states,
particularly of a driving dynamics system in a motor vehicle.
Controller 20 includes a computing element 21 that is
20   configured as a microprocessor. On computing element 21, a
computer program 22, having multitasking capability, is
executable, which is subdivided into several functionally
linked functionalities. Computer program 22 is used to control
or regulate the system according to the present invention when
25   it runs on computing element 21. Furthermore, in controller
20, means 23 are provided for the coordination of the
execution of computer program 22. Means 23 are designed as a
control program which is also executable on computing element
21. Computer program 22 and control program 23 are stored on a
30   storage element 24, which, for example, is designed as a flash
memory. In order to process computer program 22 and control
program 23, these are transmitted to computing element 21
either in total or section by section via a data connection
25. In the same manner, results of computations that were
35   carried out in computing element 21, or other data, may be

transmitted to storage element 24 in the opposite direction via data connection 25, and stored there. Control program 23 is used for carrying out the example method according to the present invention when it is carried out on computing element 21.